HKU CS Tech Report TR-2018-05

SRAME: An Attribute based Message Encryption scheme with Keyword Search and Attribute Revocation

Ruoqing Zhang[1**], Lucas Hui[2], SM Yiu[1], Gongxian Zeng[1], Jing Wen[1], and Zechao Liu[3], Na Ai[1] and Jingxuan Wang[1]

1.The University of Hong Kong, The Department of Computer Science
2. Hong Kong Applied Science and Technology Research Institute Company Limited
3. Harbin Institute of Technology Shenzhen Graduate School

Abstract. It is quite natural nowadays for data owner to outsource their business data to the cloud. The data such as business client information, electronic health record (EHR) are more inclined to shift storage and search operations to cloud servers. But there are three critical issues that need to be solved in the application process. Firstly, how to make data to be shared by different organizations in an efficient and privacy-preserving way? Secondly, whether it can still execute normal data query on those data if encrypted? Lastly but most importantly, How to guarantee the multiple-organization query is secure enough? In order to address those issues, in this paper, we adopt some technologies called Attribute-based Encryption (ABE), Complete Subset Difference (CSD) revocation and Multi-Key Searchable Encryption (MKSE) and thus propose a new architecture, called *searchable and revocable attribute-based message encryption* (SRAME). A thorough security and performance analysis shows that our design is secure and efficient. We believe our work explores a possible approach on how to build a secure multinational data sharing solution for cloud storage services.

Keywords: Attribute Based Encryption, Searchable Encryption, Attribute Revocation, Broadcast Encryption

1 Introduction

Cloud computing allows data owners to use massive data storage and ample computation capabilities at an acceptable cost. While this advantage also will bring leakage risk and uncertain control to the outsourced cloud data. To mitigate such concerns, cloud services providers always offer encryption approach before data owners store their data in the cloud. However, there exists a balance the higher of the encryption level, the lower of data usage efficiency will be. For instance, a data owner may not be able to grant usage right conventionally to data users or other recognized department or organization. He or she has to classify those identities and consider a secure sharing approach, especially for the multiple-organization scenario.

Let us make an example, Suppose Alice wants to share some encrypted data in the cloud with Bob, she needs to give him the keys of the corresponding ciphertext. To do this, Alice encrypts these keys with Bob's public key and uploads the resulting wrapped keys to a database. After downloading and decrypting them, Bob gains access to those allowed data. A user always needs to share data to multiple receivers, like other organizations, and those receivers may belong to the same group or have no relationship to each other. If Alice faces the above situation and still follows the way she transfers to Bob, the workload will be much huge and thus increases the possibility of mistake.

[0] Corresponding Author: Ruoqing Zhang, Email Address: rqzhang2@cs.hku.hk

An appropriate method is to try to find the relationship between receivers, or make receivers' identities clear enough. In other words, we hope the ready-shared data can "distinguish" the targeted receiver automatically. There are several technologies such as Identity-based Encryption [8], Attribute-based Encryption [16] have been proposed for the demand. We have investigated that Attribute-based Encryption (ABE) owns the idle "distinguishing" ability above. This asymmetrical data encryption algorithm allows users to encrypt and decrypt data based on user attributes, such as title, age, position, with flexible access control based on those attributes. We will set forth this encryption algorithm and related research later.

In addition, it would be difficult to execute keyword searching over the data owner's encrypted data in the cloud. While keyword searching is a crucial demand for cloud storage. In cloud ear, most stored data such as client information in business companies, electronic health record (EHR) in hospitals will be used at any time between different parties. Making the data stored in a public cloud and let privileged party to search and access is a general method. Obviously, the search requirement of multiple-organization sharing, this requirement becomes even harder for ensuring that each organization can search keyword without conflict and leakage. Last but the most important, there must be proved as secure as possible even if a scheme can be proposed to satisfy the above issues.

1.1 Our Contribution

We propose a cryptographic solution by which data owner can control their shared data to the targeted receivers according to encryption and access control policy. More specifically, we adopt a novel asymmetrical data encryption algorithm called ABE which can offer "distinguishing" targeted receiver capability and high efficiency.

In addition, we adopted the Multi-Key Searchable Encryption (MKSE) scheme supporting the normal data query between multiple-organization. This scheme can offer the keyword searching on the document encrypted with different organization's keys. Combining ABE and MKSE scheme together, the issues of secure data sharing and normal keyword query between different organization can be solved very well. We also give detailed correctness and security analysis to make the scheme more convincing.

For further targeted receivers identity management after adopting ABE, owing to that the receivers' identities are tagged with their attributes, we introduce the Subset Difference (SD) framework and Complete Tree Subset Difference (CSD) scheme to realize the attribute revocation process. According to the attribute revocation or adjunction mechanism, the ABE encryption algorithm can be more able to meet actual needs.

1.2 Related Work

To the best of our knowledge, there is no solution to be satisfactory for what we want to achieve. Nevertheless, we briefly review the relevant techniques below.

Attribute Based Encryption Attribute-based Encryption (ABE) is an expansion of public key encryption proposed in 2005 [27]. This encryption algorithm enables fine-grained access control policy for encrypted data. The sender can effectively determine which kinds of receiver's attributes meet the policy requirement. The access-control

policy is a boolean formula which consists of attributes type and "AND", "OR" basic operations, e.g. This formula {"Bob" OR ("IACR board" AND (January 1, 2011 $\leqslant$ "date" $\leqslant$ December 31, 2014))} defines only Bob or the IACR members who belong to the above period have the permissions to access the encrypted message. The policy thus can "distinguish" the unauthorized members at the same time. There are two formulations of ABE: Key Policy (KP)-ABE and Ciphertext Policy (CP)-ABE, depending on whether the attributes are inserted in the ciphertext or whether the access-policy is inserted in the ciphertext. In KP-ABE [16], attributes are used to annotate the ciphertexts, and the access formulas over these attributes are ascribed to users' secret keys. Conversely, CP-ABE [33], the access policy is located in the ciphertext. Some different attributes are inserted to a private key to stand its user.

As so far, the research phase of ABE can be divided into three phase: *classical ABE phase* (2005 - 2009), *dual-system ABE phase* (2010 - 2016), and *fast ABE phase* (2017). In *classical phase*, the prime ABE designings (e.g. [4], [16], [20]) make use of a double of multiplicative cyclic groups ($\mathbb{G}_0$, $\mathbb{G}_1$) of prime order, and a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ as basis. What's more, this phase researchers always adopt threshold secret sharing schemes to construct a tree structure as the access policy. Subsequently, the *dual-system ABE* structure ([10], [9]) was devised further, the dual system groups contain a triple of abelian groups ($\mathbb{G}_0$, $\mathbb{G}_1$, $\mathbb{G}_T$) of composite order, and a non-degenerate bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_1 \to \mathbb{G}_T$. The dual-system groups satisfy some properties such as subgroup indistinguishability, associativity, and parameter-hiding. In this phase, the vast majority of ABE scheme deployed a linear secret sharing scheme (LSSS) as standard access policy structure. What's more, other functions for ABE such as outsourcing computation [17], multi-authority [11] were also got more attention. However, the deployment of ABE algorithm system is very slow. In early ABE scheme, there is one property that both the ciphertext size and decryption time grow with the size of the access structure policy. That means when the number of system attributes increase, the decryption operation of ABE algorithm will become heavier. Therefore, researchers return back to focus the optimization of the algorithm itself and hope to improve its actual performance, In 2017, a fast ABE algorithm [1] was proposed with redesigned pairing construction, and the decryption speed thence got greatly improved. Here we give a comparison in Fig.1 and Fig.2 to show this performance improvement. Owing to this perfect efficiency we adopt this scheme, AC17 scheme, to solve the encrypted data secure sharing problem.

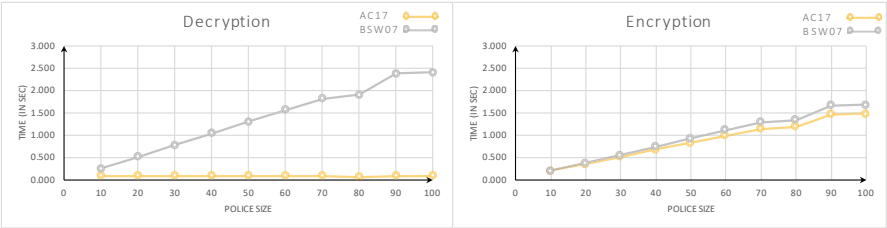


Fig. 1. ABE Schemes Encryption and Decryption Comparison[1]

[1] Note that this is a reproducibility comparison implementation about the AC17 ABE [1] and BSW07 ABE scheme [4] . All running times we tested on a VMware Virtual machine Ubuntu (12.04) system

There are indeed some schemes that have been proposed, which adopted ABE to encrypt data and made keyword searching on data. But their designing is still unsatisfied. Firstly Sun and Li [30] proposed the ciphertext policy attribute-based keyword search scheme with user revocation feature. then Han and Hu et al. [18] introduced a general transformation from the KP-ABE scheme to the ABE scheme with the keyword search. Deng and Zhou et al. [14] gave a searchable encryption scheme to support multi-user level, but the sharing feature in their scheme is not what ABE owns. Thus we want to seek a method to make an ABE scheme can have a keyword search function.

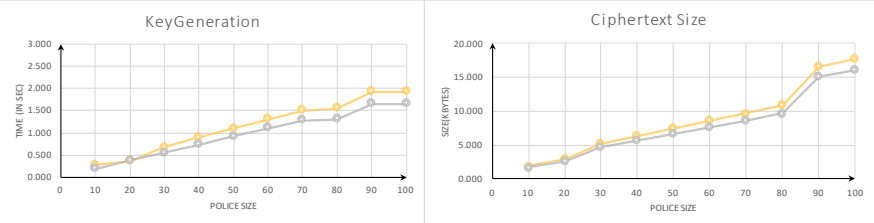


Fig. 2. ABE Schemes Key Generation and Ciphertext size Comparison[2]

Keyword Search over Encrypted Data In the cloud computing setting, it is natural for the user to not only store data after encryption but also make keyword searching on it. Existing solutions for keyword-based search over encrypted data can be classified into two categories: searchable encryption in the symmetric-key setting (e.g. [29], [13], [21]), and searchable encryption in the asymmetric-key setting (e.g. [7], [3]). In these research work, the data owner generates some tokens that can be used by searching on the encrypted data. Several variants have been proposed to support complex function (e.g. [38], [14]) such as verifying the search operation and multi-user operation. What's more, many novel cryptography ideas are introduced to help this problem such as homomorphic encryption [26], thus in [36], the researcher adopts CryptDB homomorphic encryption structure to solve the encrypted electronic health record under ABE. Well, their designing is only a framework but the detail is not clear.

All these solutions do not solve the issues studied in the present paper because (1) some solution require interactions between data users and data owners to grant search capabilities, (2) Most solution apart from [14] rare consider the keyword search operation between multi-organization. In [24], a multi-key searchable encryption based on bilinear pairing is proposed and this strategy can solve the multi-organization issue. The data owner thus can allow the users those who can share data to execute keyword search.

Attribute Revocation in ABE Key revocation is a notoriously tricky issue in the cryptosystem, and ABE is no exception. Thus in ABE schemes, the key revocation

with a 2.4 GHz Intel Core i5 processor and 5GB RAM. These two schemes are implemented in Python 3.2.3 using the Charm framework [2].

[2] The performance metrics evaluated are key generation time, ciphertext size, encryption and decryption time. Among them, the decryption time is more crucial owing that it determines actual performance in devices. We set access policies with type $'Attribute1 \cap Attribute2 \cap Attribute3 \cdots AttributeN'$ and thus the policy is of size N. We test those two CP-ABE schemes against policies and attribute sets of size 10, 20, $\cdots$, 100.

issue can be transformed into the attribute revocation problem, which user's secret key contains some attributes. There is two level of revocation mechanism in ABE, one is *user revocation*, like deleting user directly. This level is sample but crude. The other one is *attribute revocation*, which stands that a user lost some attribute and other unrelated attributes are not affected. The revocation issues, no matter user level or attribute level, are the basic features for user identity management in ABE cryptographic schemes. Actually, users' possession of a particular attribute is in dynamic. Thus attribute level revocation is more useful but harder than user attribute revocation.

Many researchers have been studying these topics for a long time. First, user revocation capability for data outsourcing systems appeared in an attribute-based access control scheme using CP-ABE [19]. Following, Zhang et al. [35] proposed a cloud-based access control scheme with user revocation and attribute update (almost can be seen as attribute revocation) in the context of ABE. Specifically, they defined user revocation in the identity-based setting and avoided conflict with their attribute-based design. In a different perspective, Chow [12] generalized the property of ABE and proposed a framework to transform any pairing-based single-authority ABE scheme into multi-authority ABE schemes with attribute-level revocation and efficient outsourced decryption. While their revocation designing is not easy to split from their complicated multi-authority designing. Zhang and Hui et al. [37] implemented a practical CP-ABE scheme which supports attribute revocation as well as other features such as outsourcing computation, traceability. Owing to that the outsourcing decryption feature, the user's key has to be exchanged twice or more between users and proxy server, thus the scheme in [37] is even complicated instead. Yamada et al. [34] combine identity-based revocation and ABE together, but their designing is just a framework concept's combination, without a particular scheme explanation as proof. That's the motivation that we want to seek enough independent and well-studied mechanism to support the attribute revocation issue in ABE.

1.3 Organization and System Model

The rest of this paper is organized as follows. After the system model presentation, Section 2 introduces the necessary technical preliminaries and main building block in the scheme. Section 3 enumerates the relative security requirements. Later we give a concrete construction for our scheme and the correctness analysis in Section 4. The relative scheme's security analysis is displayed in Section 5. At last, we state our conclusion and future work in Section 6.

Then we describe our scheme system model in Fig 3 for better understanding, there are five parties in the whole system: **Trusted Key Authority (TKA)**, the only party that is entirely trusted by other. It undertakes the key-related assignment. **Data Owner** plays as a sender and perform encryption operation for message and keyword; **Central Server**, controlled by **Central Server Manager**, are provided by a cloud storage provider service. The encrypted message and relative keywords are stored on it. Central server manager is in charge of administering the keys. The **User** act as a message receiver.

2 Technical Preliminaries

In this section, we present the necessary technical preliminaries in detail for instantiating our scheme, which includes the fundamental bilinear pairing knowledge primitives

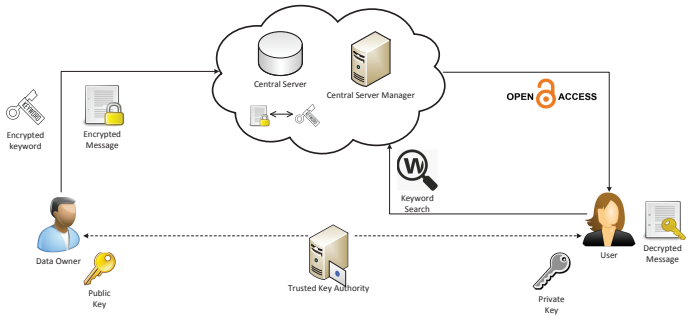


Fig. 3. System model of our scheme

and necessary cryptographic building blocks: ABE, Complete Tree Subset Difference (CSD) scheme and Multi-Key Searchable Encryption (MKSE) scheme.

2.1 Bilinear-map and Assumption

As so far, bilinear-map pairing is a fundamental mathematical structure for designing ABE and MKSE schemes. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups with the same prime order p, a bilinear pairing is defined as a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with the following properties where g_1, g_2 and g_T are generators of $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$:

1. Bilinear: $e(R^a, Q^b) = e(R, Q)^{ab}$ for any $R \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$.
2. Non-degenerate: There are $R \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$ such that $e(aR, bQ) = e(R, Q)^{ab}$.

Generally, there are three types of pairing: Type-I: $\mathbb{G}_1 = \mathbb{G}_2$. Type-II: $\mathbb{G}_1 \neq \mathbb{G}_2$ but there is an efficiently computable homomorphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$. Type-III: $\mathbb{G}_1 \neq \mathbb{G}_2$ but there are no efficiently computable homomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$. In [15], the researchers claim that Type-III pairing is the best choice for public key crypto protocols, owing to that it offers good performance and strong security guarantees. Thus we adopt Type-III pairing throughout the paper if there are no special claims. Under the Type-III pairing, the AC17 ABE scheme is provably secure under the decisional linear assumption (DLIN) which be defined as follows and is implemented based on k-linear family of assumption [28].[3]

Definition 1. *An asymmetric pairing group satisfies the decisional linear assumption (DLIN) if for all PPT the adversary $\mathcal{A}$,*

$$Adv^{\mathcal{A}}_{DLIN}(\lambda) = |Pr[\mathcal{A}(1^\lambda, par, D, T_0) = 1] - Pr[\mathcal{A}(1^\lambda, par, D, T_1) = 1]|$$

is negligible in λ, where $par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, g_T)$, g_1, g_2, g_T are the generators for $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ and p is prime order, $\mathbb{G}_1 \times \mathbb{G}_2 = \mathbb{G}_T$. $a_1, a_2 \leftarrow_R \mathbb{Z}_p^*$, $s_1, s_2, s \leftarrow_R \mathbb{Z}_p$, $D = (g_1^{a_1}, g_1^{a_2}, g_2^{a_1}, g_2^{a_2}, g_1^{a_1 s_1}, g_1^{a_2 s_2}, g_2^{a_1 s_1}, g_2^{a_2 s_2})$, $T_0 = (g_1^{s_1+s_2}, g_2^{s_1+s_2}), T_1 = (g_1^s, g_2^s)$.

[3] The researchers claim that AC17 scheme can work for which $k \geq 2$ and the security would follow from the corresponding assumption. But the code implementation only considers which $k = 2$. Those interested readers can refer to the [1] for more description.

2.2 Access Structure Policy

In ABE setting, the access structure policy is in terms of **Boolean formula**, to combine the target attributes with "AND", "OR" gates. In practice, boolean formula can be transformed to a function called **monotone span programs** (MSP) (also called linear secret sharing schemes). An MSP is given by a matrix $\mathbb{A}$ of $n_1 \times n_2$. and a mapping $\pi : \{1, \cdots, n_1\} \to \mathcal{S}$ which $\mathcal{S}$ is the attribute set. Therefore a boolean formula F can be transformed to a MSP $(\mathbb{A}, \pi)$ that each row of $\mathbb{A}$ corresponds to an input in F and the number of columns is same as the number of "AND", "OR" gates in F. Each entry in $\mathbb{A}$ is either a 0, 1, or -1.

Here we can show its mathematical expression. Give the attribute set $\mathcal{S}$ and $I = \{i|\pi(i) \in \mathcal{S}, i \in \{1, \cdots, n_1\}\}$ be the rows set in $\mathbb{A}$, $(\mathbb{A}, \pi)$ is said to accept $\mathcal{S}$ if there exists a linear combination of rows in I that gives $(1, 0, \cdots, 0)$. There should exist constants $\{\gamma_i\}_{i \in I}$ to make

$$\sum_{i \in I} \gamma_i (\mathbb{A})_i = (1, 0, \cdots, 0), \tag{1}$$

where $(\mathbb{A})_i$ is the ith row of $\mathbb{A}$. In fact the 1 in $(1, 0, \cdots, 0)$ can be seen as the shared secret value in linear secret sharing schemes.

2.3 Attribute Based Encryption

ABE be a secure algorithm and it contains: (**ABE-Setup**, **ABE-KeyGen**, **ABE-Encrypt**, **ABE-Decrypt**), where **ABE-Setup** is to initialize the system parameter and master key, **ABE-KeyGen** is to generate credentials for users, **ABE-Enc** is to encrypt the data with the access policy and **ABE-Dec** is to decrypt correspondingly.

2.4 Complete Tree Subset Difference Broadcast Encryption

The Complete Tree Subset Difference (CSD) method is essentially a tree-based key revocation method that falls under the Subset Cover (SC) framework. In NNL01 scheme [23], a tree-based Subset Difference (SD) method was proposed and then been seen as one of the most popular broadcast encryption scheme. The Subset Difference method contains two subject: Complete Tree Subset Difference (CSD) method and Subset Difference (SD) method. In [5], the researchers made detailed analytics and claimed that CSD and SD method almost have the same performance while CSD can avoid dummy users issue in system. Therefore, we choose the CSD method as the revocation mechanism to support attribute revocation property. Here we explain the Subset Cover framework at first and then give a short introduce for CSD method.

Subset Cover Framework The Subset Cover (SC) revocation framework assumes a *center* broadcasts an encrypted message M to a set $\mathcal{N}$ of n users ($|\mathcal{N}| = n$). This user set $\mathcal{N}$ contains all the possible recipients. A subset $\mathcal{R}$ of users $\mathcal{N}$ are revoked. SC framework's aim is that using a broadcast encryption algorithm, the *center* can ensure any user belonging to the set $\mathcal{N} \backslash \mathcal{R}$ can correctly decrypt the message M, while any coalition of users belonging to the set $\mathcal{R}$ should not be able to decrypt M successfully.

Definition 2. (Subset Cover). *A subset cover* (SC) *framework for a user set* $\mathcal{N}$ *consists of four PPT algorithms* **SC-Setup**, **SC-Assign**, **SC-Cover**, **SC-Match**,

which are defined as follows:
SC-Setup($\mathcal{N}$). *The setup algorithm takes as input the user set $\mathcal{N}$ and outputs a collection S of subsets $\{S_1, \cdots, S_w\}$ where each $S_j \in \mathcal{N}$, thus $\mathcal{N} = \bigcup_{j=1}^{w} S_j$. A long-lived key L_j is assigned to each subset S_j.*
SC-Assign(S,u). *The assigning algorithm takes as input the collection S and a user $u \in \mathcal{N}$, and outputs a piece of secret information I_u that is associated with the user u which belongs to a subset S_j.*
SC-Cover(S,$\mathcal{R}$). *The covering algorithm takes as the collection S and a revoked set $\mathcal{R} \in N$ of users, and generates a covering set $CV_{\mathcal{R}} = \{S_{i_1}, \cdots, S_{i_h}\}$, where each $S_{i_j} \in S$. Actually the covering set $CV_{\mathcal{R}}$ is a partition of the non-revoked users $\mathcal{N}\backslash\mathcal{R}$ into disjoint subsets $S_{i_1}, \cdots, S_{i_h}$ where $\mathcal{N}\backslash\mathcal{R} = \bigcup_{j=1}^{h} S_{i_j}$. The size $\boldsymbol{h}$ is named as the header length (later we will see why).*
SC-Match($CV_{\mathcal{R}}$, I_u). *The matching algorithm takes as input a covering set $CV_{\mathcal{R}}$ and private set I_u of a user u. It outputs (S_{i_k}, I_u) such that $S_{i_k} \in CV_{\mathcal{R}}$, $u \in S_{i_k}$ and I_u,or the algorithm outputs $\perp$.*

The correctness of SC framework is defined as follows: For all S generated in **SC-Setup**, all I_u generated by **Assign**, and any $\mathcal{R}$, it is required that:

- If $u \notin \mathcal{R}$, then **SC-Match(Cover(S,$\mathcal{R}$), I_u)** $= (S_{i_k}, I_u)$ for $S_{i_k} \in CV_{\mathcal{R}}$ and $u \in S_{i_k}$.
- If $u \in R$, then **SC-Match(Cover(S,$\mathcal{R}$), I_u)** $= \perp$.

What's more, SC utilizes two algorithms to achieve encryption: F_K and E_L.

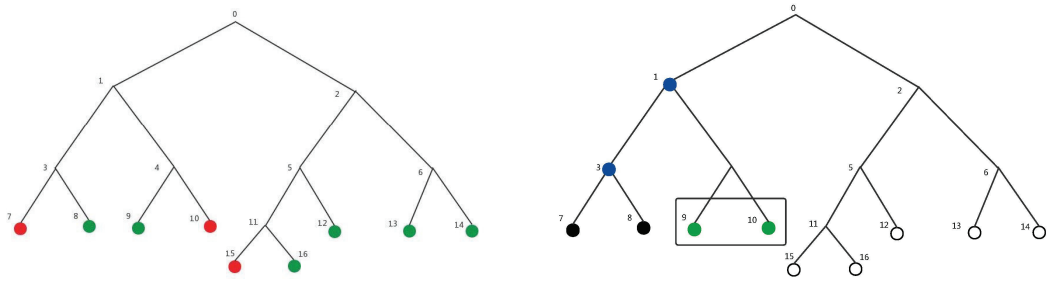
- $F_K : \{0,1\}^* \mapsto \{0,1\}^*$, is used to encrypt the message $\mathcal{M}$. Choosing a key K fresh for every $\mathcal{M}$ as *session key*. In this paper we select attribute based encryption as F_K.
- $E_L : \{0,1\}^l \mapsto \{0,1\}^l$ is used for delivering session keys to the receivers. i.e. encrypting the K with a long-lived key L_j corresponding to the subset S_j of users. A simple implementation method here is to make E_L be a block cipher like AES.

Therefore, in order to broadcast the message $\mathcal{M}$ under SC framework defined above, the *center* will choose a session key K and encrypts $\mathcal{M}$ as $F_K(\mathcal{M})$. After knowing the privileged user set $\mathcal{N}\backslash\mathcal{R}$, the *center* finds a covering set $CV_{\mathcal{R}} = \{S_{i_1}, \cdots, S_{i_h}\}$ and lets long-lived keys $L_{i_1}, \cdots, L_{i_h}$ are assigned to each subset in $CV_{\mathcal{R}}$. Then the *center* encrypts K with each of these keys L_{i_j} and thus the session key has to be encrypted h times. Those h encrypted session keys are sent along with $F_K(\mathcal{M})$ as a *header* for the encrypted message. Here we are going to refer to the size h as the *header length*. The final broadcast ciphertext to users is

$$\left\langle \underbrace{i_1, i_2, \ldots, i_h, E_{L_{i_1}}(K), E_{L_{i_2}}(K), \ldots, E_{L_{i_h}}(K)}_{header}, F_K(\mathcal{M}) \right\rangle.$$

Upon receiving the ciphertext, the user can first search through the list of $\{i_1, i_2, \ldots, i_h\}$, to find an index i_k such that $u \in S_{i_k}$, i.e. the **SC-Match** algorithm in SC framework outputs (S_{i_k}, I_u) when $u \notin \mathcal{R}$ (If such an index cannot be found, the user is actually revoked, which $u \in \mathcal{R}$). This user then gets the corresponding long-lived key L_{i_j} from I_u and utilizes L_{i_j} to decrypt the related encrypted session key, $D_{L_{i_k}}(E_{L_{i_k}}(K)) = K$. Finally the user can decrypt message $\mathcal{M}$ from $F_K(\mathcal{M})$ correctly under the session key.

Complete Subset Difference (CSD) method CSD method is one instance for SC framework. Compared with the SD method which assume users number to be a power of two, CSD method can accommodate any arbitrary number of users. Each user corresponds to a leaf node in a complete binary tree. As the SubFig I in Figure 4, A complete binary tree T^0 is a non-full complete subtree with 9 leaves to cover the users. Privileged users are marked in green and the revoked users are marked in red. $S_{i,j}$ stands the set of users in the subtree T^i but not in T^j, as $T^i \setminus T^j$. Each set $S_{i,j}$ in S has to be assigned a long-lived key $L_{i,j}$. A subset difference subset example is shown in SubFig II, Figure 4. What's more, CSD method also contains cover-finding algorithm, subset key assignment algorithm. etc. Owing to limited page here we do not expand too much. In [6] the section 3 has a more clear description.



SubFig. I. A complete binary tree with 9 users as leaves

SubFig. II. The subset difference subset $S_{1,3}$ which includes leaves under node 1 but not under 3: i.e. $S_{1,3} = T^1 \setminus T^3 = \{9, 10\}$

Fig. 4. Complete Tree Subset Difference Method

2.5 Multi-Key Searchable Encryption

Multi-Key Searchable Encryption (MKSE) is a searchable encryption framework that enables keyword search over data encrypted with different users. This scheme aims to allow a client to provide a search token to the server, but still allows the server to search for that token's word in documents encrypted with different keys. Here we give a short description and further detail can be found in [24].

In MKSE framework, there are a set of users, a server and many documents. The server stores encrypted documents. Each user has access to a subset of the documents. A user can create a document and the give access to other users to the document by giving them the decryption key of the document. (*This is very similar to the ABE's scneries*). Given the total number of words to search T, The information pieces to the server is $O(n+T)$, n deltas and T tokens. Those delta and token are used for keyword searching. The MKSE scheme also relies on bilinear pairings as the claim in section 2.1. Here we assume that all keywords are of the same length with l bits. The scheme uses hash functions $H : \{0,1\}^* \to \mathbb{G}_1$ and $H_2 : \{0,1\}^n \times \mathbb{G}_T \to \{0,1\}^{l+n}$. Here is the formal definition of MKSE scheme.

Definition 3. *A multi-key search scheme consists of seven algorithms:* **MK-Setup**, **MK-KeyGen**, **MK-Delta**, **MK-Token**, **MK-Enc**, **MK-ReToken**, **MK-Match** *which are defined as follows:*

MK-Setup$(1^{\mathcal{K}}) \rightarrow$ **parameter.** *This setup algorithm return the system parameters* $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2, g_T)$.
MK-KeyGen(parameter) $\rightarrow k$. *The key generation algorithm takes system parameters as input and outputs a secret key* k. *This secret key could be a key for a user or a document.*
MK-Delta$(k_1,k_2) \rightarrow \Delta$. *This algorithm takes two keys* k_1 *and* k_2 *and outputs a delta* $\Delta = g_2^{k_2/k_1} \in \mathbb{G}_2$.
MK-Token$(k_1, w) \rightarrow tk$. *The token algorithm takes a key* k_1 *and a word* w *as input, it outputs a search token* $tk = H(w)^{k_1} \in \mathbb{G}_1$
MK-Enc$(k_2, w) \rightarrow c$. *The encryption algorithm also takes a key* k_2 *and a word* w *as input, draws a random* r *from* $\mathbb{G}_T$. *It firstly computes* $c' = H_2(r, e(H(w), g_2)^k)$, *and then outputs an encrypted word* $c = (r, c') = (r, H_2(r, e(H(w), g_2)^{k_2}))$.
MK-ReToken$(tk, \Delta) \rightarrow tk'$. *The retoken algorithm takes as input a token* tk *and a delta* Δ, *then outputs a search token* $tk' = e(tk, \Delta) \in \mathbb{G}_T$.
MK-Match$(tk', c) \rightarrow 0$ *or* 1. *The match algorithm takes as input a token* tk' *and a encrypted word* c, *return 1 if* $H_2(r, tk') = c'$ *or 0 otherwise.*

The correctness of MKSE has the following requirements.

- For all keywords w, **MK-Match**(tk', c) returns 1 with probability overwhelmingly close to one if tk' is a search token for w and c is an encryption of w.
- For the keywords $w \neq w'$, **MK-Match**(tk', c) returns 0 with probability overwhelmingly close to one if tk' is search token for w' but c is an encryption of w.

3 Security Requirement

Now we give the security requirement to our scheme. Owing to that we combine ABE, revocable broadcast encryption and searchable encryption schemes together, so the general security requirements should cover these three dimensions.

3.1 IND-CPA Security

Generally, we call an ABE crypto scheme is secure *against chosen plaintext attacks* (CPA) if no group of colluding users can distinguish between encryption of msg_0 and msg_1 under an access structure of their choice as long as no member of the group is authorized to decrypt on his/her own. On the other hand, a weaker notion called *selective* security only prevents CPA attacks when $\mathbb{A}^*$ is chosen even before the system is deployed, which is unlikely to happen in real practice.

Suppose a CP-ABE scheme is marked as II, the adaptive security game for II is defined as a game $\text{Expt}_{\text{II},\mathcal{A}}^{\text{IND-CPA}}$ between a challenger $Chal$ and an adversary $\mathcal{A}$. The game process is as follows.

1. **Setup**. $Chal$ runs to generate system parameter, pk, msk and then $Chal$ gives parameters and pk to $\mathcal{A}$.
2. **Phase 1**. $\mathcal{A}$ sends a set of attributes S. $Chal$ then runs KGen to obtain a key, which is returned to $\mathcal{A}$. This step always is repeated as many times as $\mathcal{A}$ desires.
3. **Challenge**. $\mathcal{A}$ submits two equal length messages m_0 and m_1, and a challenge access structure $\mathbb{A}^*$ to $Chal$. Note that none of the attributes set S satisfies the access structure. $Chal$ then spins a random number β, and encrypts m_β under $\mathbb{A}^*$. The ciphertexts c^* is finally given to the adversary $\mathcal{A}$.

4. **Phase 2**. This phase is almost as same as the **Phase 1**. $\mathcal{A}$ The only restriction is for those attributes in S expect the used in **Phase 1**, none of them satisfy the access structure corresponding to the challenge $\mathbb{A}^*$. This step always is repeated as many times as $\mathcal{A}$ desires.
5. **Guess**. The adversary outputs a guess β' of β, and can claim success if $\beta' = \beta$.

   The advantage of $\mathcal{A}$ in the game is defined as $Adv_{\Pi,\mathcal{A}}^{\text{CP-ABE}} = |Pr[\beta' = \beta] - \frac{1}{2}|$.

Definition 4. *A CP-ABE scheme* Π *is called adaptively secure if for all PPT adversaries $\mathcal{A}$, the advantage*

$$Adv_{\Pi,\mathcal{A}}^{\text{CP-ABE}}(\lambda) = |Pr[\text{Expt}_{\Pi,\,\mathcal{A}}^{\text{IND-CPA}}(\lambda) = 1] - \frac{1}{2}|$$

is negligible in λ, which λ is the security parameter.

3.2 Backward and Forward Security.

For a general revocation framework, a basic security requirement is satisfying *backward security* and *forward security*. According to original definition in [22]. *backward security* means that any user who comes to hold an attribute should be prevented from accessing the plaintext of the previous data exchanged before he holds the attribute. Also, *forward security* means that any user who drops an attribute should be prevented from accessing the plaintext of the following data exchanged after he drops the attribute. These two security requirements are much significant in the real application.

3.3 Keyword Access Pattern Leakage in MKSE.

A security attack for MKSE scheme is leakage named keyword access pattern. The researchers in [31] pointed presented such an attack setting, which means a malicious user A authorized by the data owner collides with the server. Therefore, A and the server can learn the plaintexts of the encrypted keywords of all the files that A can access to. Thus, if another user B queries to these files and successfully finds the keyword w, then user A and server can also know plaintext of w.

4 Scheme Construction

This section gives a scheme called *searchable and revocable attribute-based message encryption* (SRAME) which means it covers searchable encryption, attribute revocation and attribute-based message encryption. Then we give the formal algorithms syntax and concrete construction for this scheme.

4.1 Algorithm Definition

The SRAME scheme can be divided into these steps:

– **SRAME.Initiation**. The initiation algorithm is aimed at the initial parameters generation for the following steps. It includes the generating groups parameters and assigning each user in related attribute class. Takes the user set $\mathcal{N}$ and attribute set $\mathcal{S}$ as input, this algorithm outputs the security parameters $Param$ and the attribute classes G_{λ_i} related to each attribute λ_i. For every attribute classes G_{λ_i},

the **SC-Setup**($\mathcal{N}$) and **SC-Assign** in SC framework will be executed.
(*Noted that the execution outcome means that each attribute class G_{λ_i} will be associated to a user subset collection $\{S_{i_1}, \cdots, S_{i_w}\}$, and a long-lived key L_w is assigned to each subset S_{i_w}. The user needs secret information I_u to deduce long lived key*)

- **SRAME.Setup**. Given the security parameters $Param$, it outputs a search key k, public key pk and master secret key msk. the pk and msk are used to cover ABE computation.
- **SRAME.Delta**. Given k, it runs as **MK-Delta** in MKSE scheme and outputs Δ.
- **SRAME.Token**. Given a search key k_1 and a search keyword w, it runs as the **MK-Token** and outputs a token tk.
- **SRAME.KeyGeneration**. This algorithms takes msk and attribute set $\mathcal{S}$ as input and outputs user secret key sk, the sk's function as the same in ABE.
- **SRAME.Encryption**. This algorithm will be divided into three pars: the search key encryption, the search keyword encryption and the message encryption. Given a new search key k_2, the public key pk, a message $\mathcal{M}$, and an access structure $\mathbb{A}$ as input, firstly the algorithm running **ABE-Enc** to encrypt the search key k_1 as c_{k_1} under the policy $\mathbb{A}$. Then the algorithm continues to run **MK-Enc** for the search keyword w encryption under k_2, and outputs an encrypted keyword c_w. The message is also encrypted under the $\mathbb{A}$ by running **ABE-Enc**, and produces the ciphertext ct. Finally this whole encryption will output a $\mathrm{CT} = \{c_{k_1}, c_w, ct\}$.
(*Noted that even if the search key k_1 and the message $\mathcal{M}$ are encrypted respectively with the same policy, we still could not combine the two encryption togother, i.e.* **ABE-Enc**($k_1|\mathcal{M}$). *The reason is considering only the authorized user can make valid searching for the ciphertext. Separating into two ABE encryption parts can avoid the potential message-leakage on the server.*)
- **SRAME.ReEncrypt** & **Broadcast**. This algorithm is a random algorithm that takes as input the ciphertext CT and the set of $\mathcal{R}$ of users that should be revoked. If the attribute classes appear in $\mathbb{A}$, it re-encrypts CT_{re} for the attributes; else, returns $\perp$. Specifically, it exports a re-encrypted ciphertext CT_{re} that is broadcast to all non-revoked receivers. The **SC-Cover** and **SC-Match** are executed in it.
- **SRAME.Decrypt**. The decryption algorithm takes as input the ReEncrypted ciphertext CT_{re}, and a user secret key sk. This algorithm contains two phase. The first phase is the **Header Decryption**, a non-revoked user that receives the re-encrypted ciphertext CT_{re} using its secret information I_u should produce the original ciphertext (i.e. Executing **SC-Cover** and **SC-Match**). Followed is the **Message Decryption**, to decrypt message $\mathcal{M}$ from ct. (*Noted that the search key k_1 decryption is similar to* **Message Decryption**, *owing that this phase runs* **ABE-Dec** *actually, so in later description we will use* **SRAME.Decrypt**(k_1) *to stands the k_1 decryption.*)
- **SRAME.ReToken** & **Match**. This algorithm combine **MK-ReToken** and **MK-Match** together, Given the token tk, Δ and encrypted keyword c_w, it will return 1 if the generated search token tk' and the c_w are related to w, else, returns 0.

4.2 Concrete Construction

Firstly we give a SARME construction. The construction uses hash functions $\mathcal{H}$ which $\{0,1\}^* \to \mathbb{G}$ and $\mathcal{H}_2 : \{0,1\}^n \times \mathbb{G}_T \to \{0,1\}^{l+n}$, which l stands keyword bit length and n stands delta numbers. $\mathcal{H}$ and $\mathcal{H}_2$ will be modeled as random oracle.

Here we adopt the same setting in AC17. There are two types of inputs are given in $\mathcal{H}$: inputs of the form (x, l, t) or that of the form (j, l, t). The x is a string, j is a positive integer, $l \in \{1, 2, 3\}$ and $t \in \{1, 2\}$. Thus those tow inputs can be presented as xlt and $0jlt$ for simplicity. The 0 at the beginning of the second input just means it should not be confused with the first. The inputs are appropriately encoded so that no two different tuples collide.

- **SRAME.Initiation**:
  1. Takes input 1^λ, the Initiation algorithm outputs the three different groups $\mathbb{G}$, $\mathbb{H}$, $\mathbb{G}_T$ of prime order $p = \Theta(\lambda)$ equipped with a non-degenerate efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$. It also outputs the generators g and h for $\mathbb{G}$ and $\mathbb{H}$. So the security parameters $Param = (p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, g, h)$.
  2. Let the user set $\mathcal{N} = \{u_1, \cdots, u_m\}$ and attribute set $\mathcal{S} = \{\lambda_1, \cdots, \lambda_n\}$, each user will be assigned to some attribute in system setting. So the attribute class G_{λ_i} stands a set of users who hold this attribute λ_i, just like a mapping from attribute set. Here we give an short example, suppose there are four attributes $\lambda_A, \lambda_B, \lambda_C, \lambda_D$ in the $\mathcal{S}$. For a user set $\mathcal{N}$ contains three users, the user u_1, u_2, u_3 are related to $\{\lambda_A, \lambda_B, \lambda_D\}$, $\{\lambda_A, \lambda_C, \lambda_D\}$, $\{\lambda_A, \lambda_D\}$ respectively. So attribute classes are $G_{\lambda_A} = \{u_1, u_2, u_3\}$, $G_{\lambda_B} = \{u_1\}$, $G_{\lambda_C} = \{u_2\}$, $G_{\lambda_D} = \{u_1, u_2, u_3\}$.
  3. For every attribute classe G_{λ_i}, executing **SC-Setup($\mathcal{N}$)** and **SC-Assign**. It outputs a subset collection $\{S_{i_1}, \cdots, S_{i_w}\}$ and the secret information I_u for each user in this attribute set. As for how to generate the secret information. The detailed step can be found in the CSD scheme in [6].
- **SRAME.Setup**:
  1. Select $k \leftarrow \mathbb{Z}_p$.
  2. Choose $a_1, a_2 \leftarrow_R \mathbb{Z}_p^*$ and $d_1, d_2, d_3 \leftarrow_R \mathbb{Z}_p^*$. Output

$$(h, H_1 = h^{a_1}, H_2 = h^{a_2}, T_1 = e(g, h)^{d_1 a_1 + d_3}, T_2 = e(g, h)^{d_2 a_2 + d_3})$$

     as the public key pk.
  3. Choose $b_1, b_2 \leftarrow_R \mathbb{Z}_p^*$ and output

$$(g, h, a_1, a_2, b_1, b_2, g^{d_1}, g^{d_2}, g^{d_3})$$

     as the main secret key msk.
- **SRAME.Delta**: Selects keys k_1, k_2 as input and outputs delta $\Delta = h^{k_2/k_1} \in \mathbb{H}$.
- **SRAME.Token**: Takes a key k_1 and a keyword w as input, it outputs a search token $tk = \mathcal{H}(w)^{k_1} \in \mathbb{G}$.
- **SRAME.KeyGeneration**:
  1. Choose $r_1, r_2 \leftarrow_R \mathbb{Z}_p$ and compute

$$sk_0 = (h^{b_1 r_1}, h^{b_2 r_2}, h^{r_1 + r_2})$$

  2. Based on h, b_1, b_2 from msk. For all attribute $y \in \mathcal{S}$ and $t = 1, 2$, choose $\sigma_y \leftarrow_R \mathbb{Z}_p$ and compute

$$sk_{y,t} = \mathcal{H}(1yt)^{b_1 r_1 / a_t} . \mathcal{H}(2yt)^{b_2 r_2 / a_t} . \mathcal{H}(3yt)^{(r_1 + r_2)/a_t} . g^{\sigma_y / a_t},$$

     Set $sk_y = (sk_{y,1}, sk_{y,2}, g^{-\sigma_y})$.

3. Choose $\sigma' \leftarrow_R \mathbb{Z}_p$, for $t = 1, 2$, compute

$$sk'_t = \mathcal{H}(011t)^{b_1 r_1/a_t}.\mathcal{H}(012t)^{b_2 r_2/a_t}.\mathcal{H}(013t)^{(r_1+r_2)/a_t}.g^{\sigma'/a_t}.g^{d_t},$$

Set $sk' = (sk'_1, sk'_2, g^{d_3}.g^{-\sigma'})$.

4. Therefore the user secret attribute key $sk = (sk_0, \{sk_y\}_{y\in\mathcal{S}}, sk')$. Based on the diversity between users, each user holds a different sk with specified attributes.

– **SRAME.Encryption**:

1. Choose a new key k_2. Draw a $r \leftarrow \mathbb{G}_T$, for a search keyword w. Compute $c' = \mathcal{H}_2(r, e(\mathcal{H}(w), h)^k)$, and then computes an encrypted word

$$c_w = (r, c') = (r, \mathcal{H}_2(r, e(\mathcal{H}(w), h)^{k_2})).$$

Store c_w and Δ on the server side.

2. Choose $s_1, s_2 \leftarrow_R \mathbb{Z}_p$, using pk to compute

$$ct_0 = (H_1^{s_1}, H_2^{s_2}, h^{s_1+s_2}) = (h^{a_1 s_1}, h^{a_2 s_2}, h^{s_1+s_2}).$$

Suppose the access matrix $\mathbb{A}$ has n_1 rows and n_2 columns. Then, for $i = 1, \ldots, n_1$ and $l = 1, 2, 3$, compute

$$ct_{i,l} = \mathcal{H}(\pi(i)l1)^{s_1}.\mathcal{H}(\pi(i)l2)^{s_2}.\prod_{j=1}^{n_2}(\mathcal{H}(0jl1)^{s_1}.\mathcal{H}(0jl2)^{s_2})^{(\mathbb{A})_{i,j}},$$

where $(\mathbb{A})_{i,j}$ denotes the (i, j)th element in the matrix $\mathbb{A}$. Then set $ct_i = (ct_{i,1}, ct_{i,2}, ct_{i,3})$.

3. Compute $ct' = T_1^{s_1}.T_s^{s_2}.\mathcal{M}$, output the ciphertext related to message

$$ct = (ct_0, ct_1, \ldots, ct_{n_1}, ct'),$$

where n_1 stands the specified attributes number in access matrix.

4. Execute the above three steps once again, to obtain the encrypted search key c_{k_1} under the same policy matrix $\mathbb{A}$.
5. Output the ciphertext $\mathrm{CT} = \{c_{k_1}, c_w, ct\}$. Store CT and Δ on the server side.

– **SRAME.ReEncrypt** & **Broadcast**: When the encrypted message is retrieved by another user, this algorithm are executed. Before user getting the CT, the **Central Server Manager** re-encrypts the ciphertext using a set of the member's information for the G_{λ_i} that appears in the access matrix structure, to enforce access control per each attribute class on top of the ciphertext. It runs as follows:

1. Chooses a random $K_{\lambda_i} \in \mathbb{Z}_p^*$ to each attribute λ_i from $i = 1$ to n_1. Then re-encrypt ct_i as $ct_i^{K_{\lambda_i}}$, namely, $ct_i^{K_{\lambda_i}} = (ct_{i,1}^{K_{\lambda_i}}, ct_{i,2}^{K_{\lambda_i}}, ct_{i,3}^{K_{\lambda_i}})$, Therefore

$$ct_{re} = (ct_0, ct_1^{K_{\lambda_1}}, \ldots, ct_{n_1}^{K_{\lambda_{n_1}}}, ct'),$$

Set $\mathrm{CT}_{re} = \{c_{k_1}, c_w, ct_{re}\}$.

2. In each attribute class G_{λ_i}, the server adopts the SC method, to select root nodes of the minimum cover set that can include all the unrevoked nodes.
3. Follow the cover partition method, the central server manager encrypts with long-lived keys $L_{i_1}, \cdots, L_{i_m}$ and sends the ciphertext

$$\left\langle i_1, i_2, \ldots, i_h, E_{L_{i_1}}(K), E_{L_{i_2}}(K), \ldots, E_{L_{i_h}}(K), \mathrm{CT}_{re} \right\rangle$$

4. we have claimed that the part of $(i_1, i_2, \ldots, E_{L_{i_1}}(K), E_{L_{i_2}}(K), \ldots)$ is called *header*, represented as Hdr. Once receiving a download query, the **Central Server Manager** responds with $(\mathrm{Hdr}, \mathrm{CT}_{re})$.
(*Noted that there is no need to executing* **SRAME.ReEncrypt** & **Broadcast** *algorithm for the c_{k_1}, the function of c_{k_1} is for user having permission to search, not downloading for access. Next we show how the user search the keyword.*)

– **SRAME.KeyUpdate**:

1. If user's attribute has no revocation, he update the $sk_{y,t}$ in secret attribute key sk to $sk_{re_{y,t}} = sk_{y,t}^{K_{\lambda_i}}$, in the new secret attribute key sk_{re}.

2. So the $sk_{re_{y,t}}$ is

$$sk_{re_{y,t}} = \mathcal{H}(1yt)^{b_1 r_1 K_{\lambda_y}/a_t}.\mathcal{H}(2yt)^{b_2 r_2 K_{\lambda_y}/a_t}.\mathcal{H}(3yt)^{(r_1+r_2)K_{\lambda_y}/a_t}.g^{\sigma_y K_{\lambda_y}/a_t},$$

Set $sk_{re_y} = (sk_{re_{y,1}}, sk_{re_{y,2}}, g^{-\sigma_y K_{\lambda_y}})$ for y in the attribute set.

– **SRAME.Decrypt**:

1. When a user receives the ciphertext $(\mathrm{Hdr}, \mathrm{CT}_{re})$, she first finds a_b such that $u \in S_{a,b}$. and extracts the corresponding key L_{i_j} from I_u. Using the set of induced variables stored. The user decrypts E_L to obtain the $K_{\lambda_1}, K_{\lambda_2}, \ldots, K_{\lambda_i}$ and execute the **SRAME.KeyUpdate** step.
2. If the user's attributes in sk' satisfies the LSSS $(\mathbb{A}, \pi)$ in ct, then there exists constans $\{\gamma_i\}_{i\in I}$ that satisfy the equation. Compute $num =$

$$ct'.e(\prod\nolimits_{i\in I} ct_{i,1}^{K_{\lambda_i}\cdot\gamma_i}, sk_{0,1}).e(\prod\nolimits_{i\in I} ct_{i,2}^{K_{\lambda_i}\cdot\gamma_i}, sk_{0,2}).e(\prod\nolimits_{i\in I} ct_{i,3}^{K_{\lambda_i}\cdot\gamma_i}, sk_{0,3})$$

and compute $den =$

$$e(sk'_1.\prod_{i\in I} sk_{re_{\pi(i),1}}^{\gamma_i}, ct_{0,1}).e(sk'_2.\prod_{i\in I} sk_{re_{\pi(i),2}}^{\gamma_i}, ct_{0,2}).e(sk'_3.\prod_{i\in I} sk_{re_{\pi(i),3}}^{\gamma_i}, ct_{0,3}),$$

Noted that $sk_{re\pi(i),1}$, $sk_{re_{\pi(i),2}}$, $sk_{re\pi(i),3}$ denote the first, second and third elements of $sk_{re_{\pi(i)}}$, the same for ct_0.
3. Compute num/den. It should output message $\mathcal{M}$ otherwise error symbol $\perp$.

– **SRAME.ReToken** & **Match**: When an authorized user want to search the keyword w related the encrypted message. Those steps will be executed:

1. The user runs **SRAME.Decrypt(k_1)** to get the k_1.
2. The user submits a search token tk based on k_1 and keyword w.
3. Server generate a search token $tk' = e(tk, \Delta) \in \mathbb{G}_T$ based on tk and Δ.
4. Server take the tk' and c_w as input and outputs 1 if the c_w is the encryption of w indeed. The user thus gets the correct searching outcome.

4.3 Correctness Analysis

Message Decryption Correctness We show that when $\mathcal{S}$ satisfies $(\mathbb{A}, \pi)$, the decryption recovers the correct message with probability one.

1. For $l = 1, 2, 3$,

$$\prod_{i\in I} ct_{i,1}^{K_{\lambda_i}\cdot\gamma_i} = \prod_{i\in I}[\mathcal{H}(\pi(i)l1)^{K_{\lambda_i}\gamma_i s_1}.\mathcal{H}(\pi(i)l2)^{K_{\lambda_i}\gamma_i s_2}.\prod_{j=1}^{n_2}(\mathcal{H}(0jl1)^{s_1}.\mathcal{H}(0jl2)^{s_2})^{\gamma_i K_{\lambda_i}(\mathbb{A})_{i,j}}]$$

$$= [\prod_{i\in I}\mathcal{H}(\pi(i)l1)^{K_{\lambda_i}\cdot\gamma_i s_1}.\mathcal{H}(\pi(i)l2)^{K_{\lambda_i}\gamma_i s_2}].[\prod_{j=1}^{n_2}(\mathcal{H}(0jl1)^{s_1}.\mathcal{H}(0jl2)^{s_2})^{\sum_{i\in I}\gamma_i K_{\lambda_i}(\mathbb{A})_{i,j}}]$$

$$= [\prod_{i\in I}\mathcal{H}(\pi(i)l1)^{K_{\lambda_i}\gamma_i s_1}.\mathcal{H}(\pi(i)l2)^{K_{\lambda_i}\gamma_i s_2}].\mathcal{H}(0jl1)^{s_1\sum_{i\in I}K_{\lambda_i}}.\mathcal{H}(0jl2)^{s_2\sum_{i\in I}K_{\lambda_i}},$$

based on the equality (1) in section 2.2
2. So the *num* can be transformed further as

$$num = ct'.e(\prod\nolimits_{i\in I} ct_{i,1}^{K_{\lambda_i}\cdot\gamma_i}, sk_{0,1}).e(\prod\nolimits_{i\in I} ct_{i,2}^{K_{\lambda_i}\gamma_i}, sk_{0,2}).e(\prod\nolimits_{i\in I} ct_{i,3}^{K_{\lambda_i}\gamma_i}, sk_{0,3})$$

$$= ct'.\prod_{t\in 1,2}\{e(\mathcal{H}(011t),h)^{b_1r_1s_t\sum K_{\lambda_i}}.e(\mathcal{H}(012t),h)^{b_2r_2s_t\sum K_{\lambda_i}}.e(\mathcal{H}(013t),h)^{(r_1+r_2)s_t\sum K_{\lambda_i}}.$$

$$\prod_{i\in I}[e(\mathcal{H}(\pi(i)1t)^{\gamma_i K_{\lambda_i}},h)^{b_1r_1s_t}e(\mathcal{H}(\pi(i)2t)^{\gamma_i K_{\lambda_i}},h)^{b_2r_2s_t}e(\mathcal{H}(\pi(i)3t)^{\gamma_i K_{\lambda_i}},h)^{(r_1+r_2)s_t}]\}$$

3. What's more, *dem* also can be transformed further as

$$dem = \prod_{t\in 1,2}\{e(\mathcal{H}(011t),h)^{b_1r_1s_t\sum K_{\lambda_i}}.e(\mathcal{H}(012t),h)^{b_2r_2s_t\sum K_{\lambda_i}}.e(\mathcal{H}(013t),h)^{(r_1+r_2)s_t\sum K_{\lambda_i}}.$$

$$\prod_{i\in I}[e(\mathcal{H}(\pi(i)1t)^{\gamma_i K_{\lambda_i}},h)^{b_1r_1s_t}e(\mathcal{H}(\pi(i)2t)^{\gamma_i K_{\lambda_i}},h)^{b_2r_2s_t}e(\mathcal{H}(\pi(i)3t)^{\gamma_i K_{\lambda_i}},h)^{(r_1+r_2)s_t}]\}.$$

$$\langle(\prod_{t\in\{1,2\}} e(g^{d_t}.g^{\frac{\sigma'}{a_t}}.\prod_{i\in I} g^{\frac{\gamma_i K_{\lambda_i}\sigma_{\pi(i)}}{a_t}}, h^{a_t s_t})).e(g^{d_3}.g^{-\sigma'}.\prod_{i\in I} g^{-\gamma_i K_{\lambda_i}\sigma_{\pi(i)}}, h^{s_1+s_2})\rangle,$$

the above pairing part in $\langle\ \rangle$ of *dem* can finally be computed as

$$e(g,h)^{d_1a_1s_1+d_2a_2s_1+d_3(s_1+s_2)},$$

which is equal to the $T_1^{s_1}.T_s^{s_2}$ in ct'.
4. Therefore, when the *num* is divided by *dem*, the only part left is $\mathcal{M}$. Hence the message is correctly recovered.

Attribute Revocation Correctness Here we talk about the attribute revocation issue. Revoking an attribute of a user can be seen as sending a leave request for the attribute class. Suppose that the user u_1 now drops the attribute λ_A, the central server manager selects a new K'_{λ_A} which is different from the previous attribute class key K_{λ_A}. For other attribute class without any revocation, the attribute class keys K_{λ_i} remain as the same before. Then the central server manager runs **SRAME.ReEncrypt**&**Broadcast** again to generate new CT_{re} which contains K'_{λ_A}.

When executing **SRAME.KeyUpdate** and **SRAME.Decrypt**, user u_1 will have no power to decrypt the E_L as before owing that he loses the attribute λ_A. Thus he could not obtain the new K'_{λ_A}, so u_1 cannot update his secret attribute key and does the following decryption process. For other unaffected users, they can update their secret attribute keys as normal. Therefore the attribute revocation function is realized.
(*Noted that adding an attribute for a user, i.e.* ***attribute adjunction****, is also similar to the previous revocation analysis, because attribute adjunction can also be seen as an adding request for the target attribute class.*)

Search Decryption Correctness Here we explain how the keyword searching can be executed on the ciphertext. Note that before this step, the user should get the correct k_1 from ABE. Server holds $c_w = (r, c') = (r, \mathcal{H}_2(r, e(\mathcal{H}(w), h)^{k_2}))$ and $\Delta = h^{k_2/k_1}$.

1. When the user submit a token $tk = \mathcal{H}(w)^{k_1} \in \mathbb{G}$, the server compute the search token $tk' = e(tk, \Delta) = e(\mathcal{H}(w), h)^{k_2}$.
2. The server further compute $H_2(r, tk') = H_2(r, e(\mathcal{H}(w), h)^{k_2})$. If the submit w in token is the w in c_w, the $H_2(r, tk')$ will be equal to the c' in c_w, which means **MK-Match** returns 1 with probability close to one.

5 Scheme Security Analysis

The security analysis for SRAME includes four parts: the collusion resistance ability and IND-CPA security for ABE scheme, the backward and forward security for revocation mechanism and the keyword leakage in MKSE.

5.1 Collusion Resistent

Collusion resistance is a basic required security property of any ABE system from prior work view [27], [4]. The meaning of collision resistance is that if multiple users collude, they should only be able to decrypt a ciphertext if at least one of the users could decrypt it on their own. In other words, different users cannot combine their secret keys together to decrypt a ciphertext that the colluding users should not have permission. Related theory details can be found in [4].

In our scheme, the **SRAME.KeyGeneration** algorithm generates different random values σ_y for each user. Owing to their keys are randomized, so it can not be combined for different users. When decrypting the message the attacker needs to know how to recover the pairing. In order to do this, the attacker need pair c_0 from the ciphertext with the sk_0 component from user's secret attribute key. This will result in the desired value $e(g,g)^{\alpha s}$, but blinded by some value $e(g,g)^{\sigma_y s}$. This value can be blinded out if and only if the user has the correct key components to satisfy the secret sharing scheme embedded in the ciphertext. Therefore collusion attacks will not help since the blinding value is randomized to the randomness from a particular user's secret attribute key.

5.2 IND-CPA Security for AC17 ABE scheme

Theorem 1. AC17 scheme *is adaptively secure* (**Definition 4**) *under the DLIN assumption on asymmetric pairing groups* (**Definition 1**) *in the random oracle model. Concretely, for any PPT adversary $\mathcal{A}$ making Q key queries in the IND-CPA security game, there exists a PPT adversary $\mathcal{B}$ such that*

$$Adv^{\text{CP-ABE}}_{\text{AC17},\mathcal{A}}(\lambda) = (8Q+2)Adv^{CP-ABE}_{\text{DLIN},\mathcal{B}}(\lambda) + (16Q+6)/p,$$

where p is the group order, λ is the security parameter.

Proof. The security proof of Therorem 1 proceeds via a series of hybrids. According to [1], a hybrid is the process that how the challenger $Chal$ interacts with an adversary $\mathcal{A}$. There are many hybrids and starts from the prime hybrid Hyb_0, the *zeroph* hybird. Hyb_0 is the one where $Chal$ and $\mathcal{A}$ interact according to the game $\text{Expt}^{\text{IND-CPA}}_{\text{II},\mathcal{A}}$ in

Section 4.1, and II stands for the the AC17 ABE scheme (The hash function $\mathcal{H}$ is assumed to act as a random oracle).

Then the first step in the security proof is to rewrite AC17 ABE scheme in a compact form interpreting the outputs of random oracle appropriately and using the notation defined in **Definition 1** to represent group elements. This compact form will be the first hybrid, Hyb_1. Owing to the detail of Hyb_1 is too long, interested readers can refer the section 4.1 in [1]. Then there is a sequence of hybrids which called Group-I hybrids are defined for proof. Group-I hybrids have $3Q$ hybrids from $\mathrm{Hyb}_{2,1,1}$ to $\mathrm{Hyb}_{2,3,Q}$, where Q is the number of key queries an adversary makes. These hybrids modify the key components one by one. There is another set of hybrids, called Group-II hybrids, to show that the encryption of any message is indistinguishable from the encryption fo a random message. Group-II hybrids has $3Q+2$ hybrids: $\mathrm{Hyb}_3, \mathrm{Hyb}_{4,1,1}, \cdots, \mathrm{Hyb}_{4,3,Q}$ and Hyb_5. Here we just give a short definition of those hybrids for $q = 1, \cdots, Q$, Those form in the following definition such as Normal, Normal*, P-normal*, SF* etc. stand various key modes. The clear description of those hybrids and key modes can refer the section 4.2 and Appendix C.1 in [1].

- $\mathrm{Hyb}_{2,1,q}$: Same as Hyb_1 except first $i-1$ kyes are Normal*, *ith* key is P-normal, and rest are Normal.
- $\mathrm{Hyb}_{2,2,q}$: Same as $\mathrm{Hyb}_{2,1,q}$ expect *ith* key is P-normal*.
- $\mathrm{Hyb}_{2,3,q}$: Same as $\mathrm{Hyb}_{2,2,q}$ expect *ith* key is Normal*.
- Hyb_3: Same as $\mathrm{Hyb}_{2,3,Q}$ except ciphertext is SF*.
- $\mathrm{Hyb}_{4,1,q}$: Same as Hyb_3 except first $i-1$ keys are SF*, ith key is P-normal*, and rest are Normal*.
- $\mathrm{Hyb}_{4,2,q}$: Same as $\mathrm{Hyb}_{4,1,q}$ except ith key is P-SF*.
- $\mathrm{Hyb}_{4,3,q}$: Same as $\mathrm{Hyb}_{4,2,q}$ except ith key is SF*.
- Hyb_5: Same as $\mathrm{Hyb}_{4,3,Q}$ except ciphertext is Rnd*.

Note that in all the hybrids, the random oracle is simulated in the same way as in the same way as in Hyb_1. Also, two additional hybrids $\mathrm{Hyb}_{2,3,0}$ and $\mathrm{Hyb}_{4,3,0}$ are defined to be the same as Hyb_1 and Hyb_3 respectively. Then we show some necessary lemmas in proof. The symbol $Adv^{\mathcal{A}}_{i,j}(\lambda)$ stand the advantage of an adversary $\mathcal{A}$ in distinguishing Hyb_i from Hyb_j when the security parameter is λ.

Lemma 1. *For any adversary $\mathcal{A}$, $Adv^{\mathcal{A}}_{0,1}(\lambda) = 0$.*

Lemma 2. *For all $q = 1, \cdots, Q$ and PPT adversaries $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ such that*

$$Adv^{\mathcal{A}}_{(2,3,q-1),(2,1,q)}(\lambda) \leq Adv^{\mathcal{B}}_{DLIN}(\lambda) + 1/p.$$

Lemma 3. *For all $q = 1, \cdots, Q$ and adversaries $\mathcal{A}$,*

$$Adv^{\mathcal{A}}_{(2,1,q),(2,2,q)}(\lambda) \leq 2/p.$$

Lemma 4. *For all PPT adversaries $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$*

$$Adv^{\mathcal{A}}_{(2,3,Q),3}(\lambda) \leq Adv^{\mathcal{B}}_{DLIN}(\lambda) + 1/p.$$

Lemma 5. *For all $q = 1, \cdots, Q$ and PPT adversaries $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$*

$$Adv^{\mathcal{A}}_{(4,3,q-1),(4,1,q)}(\lambda) \leq Adv^{\mathcal{B}}_{DLIN}(\lambda) + 1/p.$$

Lemma 6. *For all $q = 1, \cdots, Q$ and adversaries $\mathcal{A}$.*

$$Adv^{\mathcal{A}}_{(4,1,q),(4,2,q)}(\lambda) \leq 2/p.$$

Lemma 7. *For all adversaries $\mathcal{A}$, $Adv^{\mathcal{A}}_{(4,3,Q),5}(\lambda) \leq 2/p$.*

According to those lemmas, $Hyb_0 \equiv Hyb_1$ in Lemma 1, $Hyb_{2,3,q-1} \approx Hyb_{2,1,q}$ in Lemma 2, $Hyb_{2,1,q} \equiv Hyb_{2,2,q}$ in Lemma 3, $Hyb_{2,3,Q} \approx Hyb_3$ in Lemma 4, $Hyb_{4,3,q-1} \approx Hyb_{4,1,q}$ in Lemma 5, $Hyb_{4,1,q} \equiv Hyb_{4,2,q}$ in Lemma 6, and $Hyb_{4,3,Q} \approx Hyb_5$ in Lemma 7. For all $q = 1, \cdots, Q$, where $\equiv$ and $\approx$ stand for statistical and computational indistinguishability respectively from the point of view of an adversary. Note that the proof for the indistinguishability of $Hyb_{2,2,q} \equiv Hyb_{2,3,q}$ is omitted, because it is complete analogous to that of $Hyb_{2,3,q-1} \equiv Hyb_{2,1,q}$. Also, $Hyb_{4,2,q} \approx Hyb_{4,3,q}$ can be proved in a manner similar to $Hyb_{4,3,q-1} \approx Hyb_{4,1,q}$. The hybrids are indistinguishable irrespective of the bit β given to the challenger. In other words, none of the proofs have anything to do with the value of β. Thus, Hyb_0 (AC17 ABE scheme) is indistinguishable from Hyb_0 whether we start from β.

5.3 Backward and Forward Security

Forward security corresponds to the *attribute revocation* issue. As we analysis before, the user cannot obtain the new K'_{λ_A} if he drops the attribute, so he is prevented from accessing the CT_{re} after attribute revocation. Similarly, the *backward security* is related to the *attribute adjunction* and the analysis process is almost identical. Thus the backward and forward security requirement for revocation has been met.

5.4 Keyword Access Pattern Leakage in MKSE

A trivial solution to avoid this leakage is to provide the different keyword encryptions for every authorized user. In this way, the malicious user can only know the plaintext of his keyword encryptions. However, on the other side, this requires too much extra work for a data owner when he authorizes some files to one user. Another solution is inspired by [32]. In [32], the researchers propose secure multiple users searching schemes with two servers where they assume that two servers can not collude. But this scheme is much more complicated than MKSE. Thus, if we also adopt the Two-server idea, even malicious user can collude with one server, he cannot know what other is searching.

6 Conclusion and Future Work

We have introduced a novel cryptographic solution called searchable and revocable attribute-based message encryption (SRAME). The solution achieves the following: Data owners can use attribute-based encryption (ABE) approach to control their shared data and further identify targeted data users by the access policy. While the authorized data users can also execute keyword search operations on those shared data. This keyword search operation can be even supported between multiple-organization.

Furthermore, we introduce the Complete Tree Subset Difference (CSD) method to realize the attribute revocation function and thus can manage the receivers' identities, what makes the scheme more suitable for actual needs. The adopted ABE crypto building block can also offer considerable encryption/decryption performance. A theoretical analysis shows that our designing is secure enough.

There are a number of ways to study further about our designing. For example, for the keyword search a verifiable mechanism can be considered to introduced that forces the cloud to faithfully execute the search operation. What's more, designing suitable key management strategy is also very significant in practice for ABE scheme.

References

1. Shashank Agrawal and Melissa Chase. FAME : Fast Attribute-based Message Encryption. pages 665–682, 2017.
2. Joseph A Akinyele, Matthew D Green, and Aviel D Rubin. Charm : A Framework for Rapidly Prototyping Cryptosystems. (90):1010928, 2012.
3. Mihir Bellare, Alexandra Boldyreva, and Adam O?Neill. Deterministic and efficiently searchable encryption. In *Annual International Cryptology Conference*, pages 535–552. Springer, 2007.
4. John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *Security and Privacy, 2007. SP'07. IEEE Symposium on*, pages 321–334. IEEE, 2007.
5. Sanjay Bhattacherjee and Palash Sarkar. Complete tree subset difference broadcast encryption scheme and its analysis. *Designs, codes and cryptography*, 66(1-3):335–362, 2013.
6. Sanjay Bhattacherjee and Palash Sarkar. Complete tree subset difference broadcast encryption scheme and its analysis. *Designs, Codes, and Cryptography*, 66(1-3):335–362, 2013.
7. Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *International conference on the theory and applications of cryptographic techniques*, pages 506–522. Springer, 2004.
8. Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *Annual international cryptology conference*, pages 213–229. Springer, 2001.
9. Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system abe in prime-order groups via predicate encodings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 595–624. Springer, 2015.
10. Jie Chen and Hoeteck Wee. Dual system groups and its applications-compact hibe and more. *IACR Cryptology ePrint Archive*, 2014:265, 2014.
11. Sherman S.M. Chow. A Framework of Multi-Authority Attribute-Based Encryption with Outsourcing and Revocation. *Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies - SACMAT '16*, pages 215–226, 2016.
12. Sherman SM Chow. A framework of multi-authority attribute-based encryption with outsourcing and revocation. In *Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies*, pages 215–226. ACM, 2016.
13. Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. *Journal of Computer Security*, 19(5):895–934, 2011.
14. Zuojie Deng, Kenli Li, Keqin Li, and Jingli Zhou. A multi-user searchable encryption scheme with keyword authorization in a cloud storage. *Future Generation Computer Systems*, 72:208–218, 2017.
15. Steven D Galbraith, Kenneth G Paterson, and Nigel P Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.
16. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98. Acm, 2006.
17. Matthew Green, Susan Hohenberger, Brent Waters, et al. Outsourcing the decryption of abe ciphertexts. In *USENIX Security Symposium*, volume 2011, 2011.
18. Fei Han, Jing Qin, Huawei Zhao, and Jiankun Hu. A general transformation from kp-abe to searchable encryption. *Future Generation Computer Systems*, 30:107–115, 2014.
19. Junbeom Hur and Dong Kun Noh. Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Transactions on Parallel and Distributed Systems*, 22(7):1214–1221, 2011.
20. Luan Ibraimi, Qiang Tang, Pieter Hartel, and Willem Jonker. Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In *International Conference on Information Security Practice and Experience*, pages 1–12. Springer, 2009.
21. Seny Kamara, Charalampos Papamanthou, and Tom Roeder. Dynamic searchable symmetric encryption. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 965–976. ACM, 2012.

22. Yongdae Kim, Adrian Perrig, and Gene Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. In *Proceedings of the 7th ACM conference on Computer and communications security*, pages 235–244. ACM, 2000.
23. Dalit Naor, Moni Naor, and Jeff Lotspiech. Revocation and tracing schemes for stateless receivers. In *Annual International Cryptology Conference*, pages 41–62. Springer, 2001.
24. Ra Popa and Nickolai Zeldovich. Multi-key searchable encryption. pages 1–18, 2013.
25. Raluca A Popa and Nickolai Zeldovich. Multi-key searchable encryption. *IACR Cryptology ePrint Archive*, 2013:508, 2013.
26. Raluca Ada Popa, Catherine Redfield, Nickolai Zeldovich, and Hari Balakrishnan. Cryptdb: processing queries on an encrypted database. *Communications of the ACM*, 55(9):103–111, 2012.
27. Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 457–473. Springer, 2005.
28. Hovav Shacham. A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. *IACR Cryptology EPrint Archive*, 2007:74, 2007.
29. Dawn Xiaoding Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on*, pages 44–55. IEEE, 2000.
30. W Sun, S Yu, W Lou, Y T Hou, and H Li. Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud. *33rd IEEE Conference on Computer Communications, IEEE INFOCOM 2014*, 27(4):226–234, 2014.
31. Cédric Van Rompay, Refik Molva, and Melek Önen. A leakage-abuse attack against multi-user searchable encryption. *Proceedings on Privacy Enhancing Technologies*, 2017(3):168–178, 2017.
32. Cédric Van Rompay, Refik Molva, and Melek Önen. Secure and scalable multi-user searchable encryption. 2018.
33. Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *International Workshop on Public Key Cryptography*, pages 53–70. Springer, 2011.
34. Kotoko Yamada, Nuttapong Attrapadung, Keita Emura, Goichiro Hanaoka, and Keisuke Tanaka. Generic constructions for fully secure revocable attribute-based encryption. In *European Symposium on Research in Computer Security*, pages 532–551. Springer, 2017.
35. Peng Zhang, Zehong Chen, Kaitai Liang, Shulan Wang, and Ting Wang. A cloud-based access control scheme with user revocation and attribute update. In *Australasian Conference on Information Security and Privacy*, pages 525–540. Springer, 2016.
36. Ruoqing Zhang, Chiling Chow, and Lucas CK Hui. Crypt-ehrserver: Protecting confidentiality with attribute-based encryption and encrypted query processing. In *Pervasive Systems, Algorithms and Networks & 2017 11th International Conference on Frontier of Computer Science and Technology & 2017 Third International Symposium of Creative Computing (ISPAN-FCST-ISCC), 2017 14th International Symposium on*, pages 234–241. IEEE, 2017.
37. Ruoqing Zhang, Lucas Hui, Sm Yiu, Xiaoqi Yu, Zechao Liu, and Zoe L Jiang. A traceable outsourcing cp-abe scheme with attribute revocation. In *Trustcom/BigDataSE/ICESS, 2017 IEEE*, pages 363–370. IEEE, 2017.
38. Qingji Zheng, Shouhuai Xu, and Giuseppe Ateniese. Vabks: verifiable attribute-based keyword search over outsourced encrypted data. In *Infocom, 2014 proceedings IEEE*, pages 522–530. IEEE, 2014.